REMARKS

Please reconsider the application in view of the above amendments and the following remarks. Applicant thanks the Examiner for carefully considering this application and for courtesies extended during the Examiner Interview.

Disposition of Claims

Claims 1-6 and 11-18 were pending in this application. Claims 4, 11, and 12 are cancelled by this reply. Claims 1 and 13 are independent. The remaining claims depend, directly or indirectly, from claims 1 and 13.

Attorney Docket Number

Applicant requests that the Attorney Docket No. for this matter be changed from "0007056-0204/P6024" as indicated on the cover sheet received with this office action to "**16159/095001; P6024**."

Claim Amendments

Independent claims 1 and 13 have been amended to clarify the invention. In particular, the aforementioned independent claims have been amended to clarify that the invention involves: (i) instantiating an instance of said class in a main method, wherein the main method comprises an operator; (ii) calling a special operator of said class when the operator corresponding to the special operator is called, wherein a value is returned in response to the call to the special operator; and (iii) executng the operator using the value as input, (iv) wherein said class defines said value to return when said special operator is called. Support for this

amendment may be found, for example, on pages 10 and 11 of the instant specification. No new matter has been added by this amendment.

Applicant's Response to "Response to Arguments" Section

In responding to the Applicant's arguments filed on September 16, 2005, the Examiner asserted that the referenced invention appears to be directed to operator overloading (*see* Office Action mailed January 13, 2006, pp. 5-7). The Applicant respectfully asserts that the invention is not a form of operator overloading.

Briefly, operator overloading allows multiple instances of a single operator to be created, where each instance of the operator can take different variable types as arguments. For example, the addition operator (*e.g.*, "+") traditionally supports integer addition. However, assume that you want to allow the addition operator to support imaginary number addition. In this case, one would "overload" the addition operator (*i.e.*, would define an addition operator that supports imaginary number addition). During using, the addition operator could be used interchangeably for either integer addition or imaginary number addition, where the system includes functionality to determine the appropriate form of the addition operator to execute (*i.e.*, integer addition or imaginary number addition) based on the arguments supplied to the addition operator.

In contrast, the present invention uses an *instance* of a class as an argument (as opposed to a variable), where the instance of the class defines a special operator (*e.g.*, a foreach operator). Further, when an operator in the main method is called and the operator includes the instance of the class as an argument, the special operator defined in the class is called. As recited in the claim, when the special operator is called, the code associated with the special operator is executed and a value is returned to the operator (*i.e.*, the operator in the main method). Once the

value is received, the operator in the main method is executed using the result obtained from the special operator.

Rejection(s) under 35 U.S.C § 103

Claims 1-6 and 11-18 stand rejected under 35 U.S.C. § 103 as anticipated by "C++ Standard Library: A tutorial and Reference" by Josuttis (hereafter "Josuttis") in view of "Learning Perl," by Christiansen (hereafter "Christiansen"). Claims 4, 11, and 12 were cancelled by this reply. To the extent that the claims apply to the amended claims, the rejection is respectfully traversed.

The invention is directed to a dynamic programming language that allows operators (*e.g.*, foreach, increment, decrement, etc.) to be applied to classes (or more specifically, instances of classes). The following is an example of the operation of the invention as recited in the amended independent claims. Initially, the user defines a class such as the following class:

```
Class X {
        var value =[1,2,3,5,7]
        operator foreach() { return value;}
}
```

The user then proceeds to write a program in a dynamically typed language that uses class X. The program may include the following piece of code in the main method:

```
var x = new X ()
foreach i x {
        y=i
}
```

Without the benefit of the present invention, the execution of foreach i x {...} would cause the program to crash. However, using the constructs of the present invention, when the line foreach i x {...} is encountered, the special operator corresponding to the foreach operator is called, *i.e.,* the foreach operator defined in class X is called. In this example, the execution of foreach operator in class X returns [1,2,3,5,7]. Once [1,2,3,5,7] is received, the original foreach operator (*i.e.,* the foreach operator in the program as opposed to the foreach operator in class X) uses [1,2,3,5,7] as input and the executes the actions defined in the body of the original foreach operator (*i.e.,* y=i). The Applicant notes that the above example is merely intended to illustrate the operation of an embodiment of the invention and is not intended to limit the scope of the claims.

Turning to the claims, the Applicant respectfully asserts that neither Josuttis nor Christiansen teaches or suggests all the limitations recited in the amended claims. Specifically, as noted by the Examiner, Josuttis merely discloses operator overloading (see Office Action mailed January 13, 2006, pp. 5-7). However, from the above discussion, operator overloading is not equivalent to the invention as recited in the amended independent claims 1 and 13. In particular, operator overloading in Josuttis fails to teach or suggest:

(i) instantiating an instance of said class in a main method, wherein the main method comprises an operator;

(ii) calling a special operator of said class when the operator corresponding to the special operator is called, wherein a value is returned in response to the call to the special operator; and

(iii) executing the operator using the value as input.

Further, Christiansen does not teach that which Josuttis lacks, as Christiansen is only relied upon to teach a dynamically typed programming language. (*see* Office Action mailed January 13, 2006, p. 4).

In view of the above, neither Josuttis nor Christiansen, whether considered separately or in combination, teaches or suggests all the limitations recited in amended independent claims 1 and 13. Thus, amended independent claims 1 and 13 are patentable over Josuttis and Christiansen. Pending dependent claims are patentable for at least the same reasons. Accordingly, withdrawal of this rejection is respectfully requested.

Conclusion

Applicant believes this reply is fully responsive to all outstanding issues and places this application in condition for allowance. If this belief is incorrect, or other issues arise, the Examiner is encouraged to contact the undersigned or his associates at the telephone number listed below. Please apply any charges not covered, or any credits, to Deposit Account 50-0591 (Reference Number 16159/095001; P6024).

Dated: April 13, 2006

Respectfully submitted,

By T. Chyau Liang
for Robert P. Lord T. Chyau Liang
Registration No.: 46,479 #48,885
OSHA · LIANG LLP
1221 McKinney St., Suite 2800
Houston, Texas 77010
(713) 228-8600
(713) 228-8778 (Fax)
Attorney for Applicant